F 6

```
RRRRRRRRRRRR     TTTTTTTTTTTTTTT  PPPPPPPPPPPP        AAAAAAAAA     DDDDDDDDDDDD
RRRRRRRRRRRR     TTTTTTTTTTTTTTT  PPPPPPPPPPPP        AAAAAAAAA     DDDDDDDDDDDD
RRRRRRRRRRRR     TTTTTTTTTTTTTTT  PPPPPPPPPPPP        AAAAAAAAA     DDDDDDDDDDDD
RRR         RRR        TTT        PPP         PPP  AAA         AAA  DDD         DDD
RRR         RRR        TTT        PPP         PPP  AAA         AAA  DDD         DDD
RRR         RRR        TTT        PPP         PPP  AAA         AAA  DDD         DDD
RRR         RRR        TTT        PPP         PPP  AAA         AAA  DDD         DDD
RRR         RRR        TTT        PPP         PPP  AAA         AAA  DDD         DDD
RRR         RRR        TTT        PPP         PPP  AAA         AAA  DDD         DDD
RRRRRRRRRRRR           TTT        PPPPPPPPPPPP     AAA         AAA  DDD         DDD
RRRRRRRRRRRR           TTT        PPPPPPPPPPPP     AAA         AAA  DDD         DDD
RRRRRRRRRRRR           TTT        PPPPPPPPPPPP     AAA         AAA  DDD         DDD
RRR   RRR              TTT        PPP              AAAAAAAAAAAAAAA  DDD         DDD
RRR   RRR              TTT        PPP              AAAAAAAAAAAAAAA  DDD         DDD
RRR   RRR              TTT        PPP              AAAAAAAAAAAAAAA  DDD         DDD
RRR      RRR           TTT        PPP              AAA         AAA  DDD         DDD
RRR      RRR           TTT        PPP              AAA         AAA  DDD         DDD
RRR      RRR           TTT        PPP              AAA         AAA  DDD         DDD
RRR         RRR        TTT        PPP              AAA         AAA  DDDDDDDDDDDD
RRR         RRR        TTT        PPP              AAA         AAA  DDDDDDDDDDDD
RRR         RRR        TTT        PPP              AAA         AAA  DDDDDDDDDDDD
```

```
                                                         H 12
**FILE**ID**CTSETRT




                  CCCCCCCC  TTTTTTTTTT    SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT  RRRRRRRR    TTTTTTTTTT
                  CCCCCCCC  TTTTTTTTTT    SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT  RRRRRRRR    TTTTTTTTTT
                CC              TT      SS          EE              TT      RR      RR      TT
                CC              TT      SS          EE              TT      RR      RR      TT
                CC              TT      SS          EE              TT      RR      RR      TT
                CC              TT      SS          EE              TT      RR      RR      TT
                CC              TT        SSSSSS    EEEEEEEE        TT      RRRRRRRR        TT
                CC              TT        SSSSSS    EEEEEEEE        TT      RRRRRRRR        TT
                CC              TT              SS  EE              TT      RR  RR          TT
                CC              TT              SS  EE              TT      RR  RR          TT
                CC              TT              SS  EE              TT      RR    RR        TT          ....
                CC              TT              SS  EE              TT      RR    RR        TT          ....
                  CCCCCCCC      TT      SSSSSSSS    EEEEEEEEEE      TT      RR      RR      TT          ....
                  CCCCCCCC      TT      SSSSSSSS    EEEEEEEEEE      TT      RR      RR      TT          ....



                LL            IIIIII      SSSSSSSS
                LL            IIIIII      SSSSSSSS
                LL              II      SS
                LL              II      SS
                LL              II      SS
                LL              II      SS
                LL              II        SSSSSS
                LL              II        SSSSSS
                LL              II              SS
                LL              II              SS
                LL              II              SS
                LL              II              SS
                LLLLLLLLLL    IIIIII    SSSSSSSS
                LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

(2) 67 DECLARATIONS
(3) 298 CT_POST_SENSE - Map TSA into VMS

```
00000001  0000        RTPAD = 1                 ; Define to generate Rtpad module
          0000    .IF DF RTPAD
          0000            .TITLE  CTSETRT - RTPAD/CTERM SET CHARACTERISTICS
          0000    .IFF
          0000            .TITLE  CTDSENSE - CTDRIVER SENSE MODE PROCESSING
          0000    .ENDC
          0000            .IDENT  'V04-000'
          0000            .ENABLE SUPPRESSION
          0000    ;
          0000    ;****************************************************************************
          0000    ;*                                                                          *
          0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
          0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
          0000    ;*  ALL RIGHTS RESERVED.                                                    *
          0000    ;*                                                                          *
          0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
          0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
          0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
          0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
          0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
          0000    ;*  TRANSFERRED.                                                            *
          0000    ;*                                                                          *
          0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
          0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
          0000    ;*  CORPORATION.                                                            *
          0000    ;*                                                                          *
          0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
          0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
          0000    ;*                                                                          *
          0000    ;*                                                                          *
          0000    ;****************************************************************************
          0000    ;
          0000
          0000    ;++
          0000    ;
          0000    ; FACILITY:
          0000    ;
          0000    ;       CTERM Remote terminal protocol driver
          0000    ;
          0000    ; ABSTRACT:
          0000    ;
          0000    ;       This module is called to map a characteristics item list received
          0000    ;       from the net into VMS terminal characteristics.
          0000    ;
          0000    ; ENVIRONMENT:
          0000    ;
          0000    ;
          0000    ;
          0000    ;--
          0000    ;
          0000    ; AUTHOR: Jake VanNoy, CREATION DATE: 23-Aug-1982
          0000    ;
          0000    ; MODIFIED BY:
          0000    ;
          0000    ;
          0000    ;       V03-003 JLV0336         Jake VanNoy             28-FEB-1984
          0000    ;               Use constant names in SENSETAB table.
```

```
0000   ;
0000   ;         V03-002 JLV0288          Jake VanNoy              28-JUL-1983
0000   ;                 Update symbols that identify characteristics.
0000   ;
0000   ;         V03-001 JLV0261          Jake VanNoy              26-MAY-1983
0000   ;                 Change baud rate tables to new format.
0000   ;
0000   ;**
0000
```

```
                    0000        .SBTTL  DECLARATIONS
                    0000  ;
                    0000  ; INCLUDE FILES:
                    0000  ;
                    0000          $DCDEF
                    0000          $SSDEF
                    0000          $TTDEF
                    0000          $TT2DEF
                    0000          $TSADEF
                    0000          $UCBDEF                      ; for protocol errors
                    0000
                    0000  ;
                    0000  ; MACROS:
                    0000  ;
                    0000
                    0000  .MACRO  CHAR    LABEL
                    0000          .WORD           char_type!char_count
                    0000          .WORD           LABEL-SENSE_ROUTINES
                    0000  char_count = char_count + 1
                    0000  .ENDM   CHAR
                    0000
                    0000  .MACRO  BAUDTAB BPS
                    0000          .WORD   BPS
                    0000          .BYTE   TT$C_BAUD_'BPS'
                    0000  .ENDM   BAUDTAB
                    0000
                    0000  .MACRO  TERMTAB TERM,VALUE
                    0000          .BYTE   VALUE
                    0000          .ASCIC  /TERM/
                    0000  .ENDM   TERMTAB
                    0000
                    0000  .MACRO  SENSETAB CHAR
                    0000          .WORD           char_type!char
                    0000  .ENDM   SENSETAB
                    0000
                    0000
                    0000  ;
                    0000  ; EQUATED SYMBOLS:
                    0000  ;
                    0000
  00000000          0000  physical = ch$c_physical@8          ; 0 leftshifted 1 byte
  00000100          0000  logical  = ch$c_logical@8           ; 1 leftshifted 1 byte
  00000200          0000  cterm    = ch$c_cterm@8             ; 2 leftshifted 1 byte
                    0000
  00000000          0000  TT_BUF   = 0
  00000004          0000  TT_CHAR1 = 4
  00000008          0000  TT_CHAR2 = 8
  00000000          0000  TT_IOSB  = 0
                    0000
                    0000  ;
                    0000  ; OWN STORAGE:
                    0000  ;
                    0000
                    0000  .IF DF RTPAD
                00000000          .PSECT RTPAD,NOWRT
                    0000
                    0000          .EXTERNAL OUTBAND_NEW
```

```
                  0000  .IFF
                  0000          .PSECT $$$115_DRIVER,LONG
                  0000
                  0000  .ENDC
                  0000
                  0000  SENSE_TABLE:
                  0000
         00000000 0000  char_type = physical
         00000001 0000  char_count = 1
                  0000          CHAR    INPUT_SPEED             ; 1
                  0004          CHAR    OUTPUT_SPEED            ; 2
                  0008          CHAR    CHARACTER_SIZE          ; 3
                  000C          CHAR    PARITY_ENABLE           ; 4
                  0010          CHAR    PARITY_TYPE             ; 5
                  0014          CHAR    MODEM_PRESENT           ; 6
                  0018          CHAR    AUTOBAUD_DETECT         ; 7
                  001C          CHAR    MANAGEMENT_GUARANTEED   ; 8
                  0020          CHAR    SWITCH_CHAR_1           ; 9
                  0024          CHAR    SWITCH_CHAR_2           ; 10
                  0028
         00000100 0028  char_type = logical
         00000001 0028  char_count = 1
                  0028
                  0028          CHAR    MODE_WRITING_ALLOWED    ; 1
                  002C          CHAR    TERMINAL_TYPE           ; 2
                  0030          CHAR    TERMINAL_SUBTYPE        ; 3
                  0034          CHAR    OUTPUT_FLOW_CONTROL     ; 4
                  0038          CHAR    OUTPUT_PAGE_STOP        ; 5
                  003C          CHAR    FLOW_CHAR_PASS_THRU     ; 6
                  0040          CHAR    INPUT_FLOW_CONTROL      ; 7
                  0044          CHAR    LOSS_NOTIFICATION       ; 8
                  0048          CHAR    LINE_WIDTH              ; 9
                  004C          CHAR    PAGE_LENGTH             ; 10
                  0050          CHAR    STOP_LENGTH             ; 11
                  0054          CHAR    CR_FILL                 ; 12
                  0058          CHAR    LF_FILL                 ; 13
                  005C          CHAR    WRAP                    ; 14
                  0060          CHAR    HORIZONTAL_TAB          ; 15
                  0064          CHAR    VERTICAL_TAB            ; 16
                  0068          CHAR    FORM_FEED               ; 17
                  006C
         00000200 006C  char_type = cterm
         00000001 006C  char_count = 1
                  006C
                  006C          CHAR    IGNORE_INPUT            ; 1
                  0070          CHAR    CHAR_ATTRIBUTES         ; 2
                  0074          CHAR    CONTROL_O_PASS_THRU     ; 3
                  0078          CHAR    RAISE_INPUT             ; 4
                  007C          CHAR    NORMAL_ECHO             ; 5
                  0080          CHAR    INPUT_ESCAPE_ENABLE     ; 6
                  0084          CHAR    OUTPUT_ESCAPE_ENABLE    ; 7
                  0088          CHAR    INPUT_COUNT_STATE       ; 8
                  008C          CHAR    AUTO_PROMPT             ; 9
                  0090          CHAR    ERROR_PROCESSING        ; 10
                  0094
             FFFF 0094          .WORD   -1                      ; End of list (any negative number)
                  0096
```

```
                             0096   181 SENSE_BAUD:                                    ; Table must be in ascending order
                             0096   182             BAUDTAB 50
                             0099   183             BAUDTAB 75
                             009C   184             BAUDTAB 110
                             009F   185             BAUDTAB 134
                             00A2   186             BAUDTAB 150
                             00A5   187             BAUDTAB 300
                             00A8   188             BAUDTAB 600
                             00AB   189             BAUDTAB 1200
                             00AE   190             BAUDTAB 1800
                             00B1   191             BAUDTAB 2000
                             00B4   192             BAUDTAB 2400
                             00B7   193             BAUDTAB 3600
                             00BA   194             BAUDTAB 4800
                             00BD   195             BAUDTAB 7200
                             00C0   196             BAUDTAB 9600
                             00C3   197             BAUDTAB 19200
                   00000000  00C6   198             .LONG   0
                             00CA   199
                             00CA   200 CT$AB_TERM_TABLE::                             ; Table is in order for most likely first
                             00CA   201             TERMTAB VT100,  TT$_VT100
                             00D1   202             TERMTAB VT200,  TT$_VT200_Series
                             00D8   203             TERMTAB VT101,  TT$_VT101
                             00DF   204             TERMTAB VT102,  TT$_VT102
                             00E6   205             TERMTAB VT105,  TT$_VT105
                             00ED   206             TERMTAB VT125,  TT$_VT125
                             00F4   207             TERMTAB VT131,  TT$_VT131
                             00FB   208             TERMTAB VT132,  TT$_VT132
                             0102   209             TERMTAB VT52,   TT$_VT52
                             0108   210             TERMTAB VT05,   TT$_VT05
                             010E   211             TERMTAB VK100,  TT$_VK100
                             0115   212             TERMTAB VT173,  TT$_VT173
                             011C   213 ;           TERMTAB TT$_FT1
                             011C   214 ;           TERMTAB TT$_FT2
                             011C   215 ;           TERMTAB TT$_FT3
                             011C   216 ;           TERMTAB TT$_FT4
                             011C   217 ;           TERMTAB TT$_FT5
                             011C   218 ;           TERMTAB TT$_FT6
                             011C   219 ;           TERMTAB TT$_FT7
                             011C   220 ;           TERMTAB TT$_FT8
                             011C   221             TERMTAB LA36,   TT$_LA36
                             0122   222             TERMTAB LA12,   TT$_LA12
                             0128   223             TERMTAB LA34,   TT$_LA34
                             012E   224             TERMTAB LA38,   TT$_LA38
                             0134   225             TERMTAB LA12,   TT$_LA12
                             013A   226             TERMTAB LA100,  TT$_LA100
                             0141   227             TERMTAB LA24,   TT$_LA24
                             0147   228             TERMTAB LQP02,  TT$_LQP02
                             014E   229             TERMTAB VT55,   TT$_VT55
                         00  0154   230             .BYTE   0
                             0155   231
 4E 57 4F 4E 4B 4E 55 00'    0155   232 UNKNOWN_TT:         .ascic  /UNKNOWN/
                         07  0155
                             015D   233
                             015D   234 .IF NDF RTPAD   ; IF CTDRIVER...
                             015D   235
                             015D   236 CTP$AB_SENSEBUF::
```

```
015D   237
015D   238              .BYTE          CTP$C_MT_READ_CHAR       ; message type
015D   239              .BYTE          0                        ; flags
015D   240
015D   241  ;
015D   242  ; Item list, for now, request everything.
015D   243  ;
015D   244
015D   245  char_type = physical
015D   246
015D   247              SENSETAB       ch$c_ph_in_speed         ; INPUT_SPEED
015D   248              SENSETAB       ch$c_ph_out_speed        ; OUTPUT_SPEED
015D   249              SENSETAB       ch$c_ph_char_size        ; CHARACTER_SIZE
015D   250              SENSETAB       ch$c_ph_parity_enable    ; PARITY_ENABLE
015D   251              SENSETAB       ch$c_ph_parity_type      ; PARITY_TYPE
015D   252              SENSETAB       ch$c_ph_modem_present    ; MODEM_PRESENT
015D   253              SENSETAB       ch$c_ph_autobaud         ; AUTOBAUD_DETECT
015D   254  ;;;         SENSETAB       ch$c_ph_manage_guar      ; MANAGEMENT_GUARANTEED
015D   255  ;;;         SENSETAB       ch$c_ph_switch1          ; SWITCH_CHAR_1
015D   256  ;;;         SENSETAB       ch$c_ph_switch2          ; SWITCH_CHAR_2
015D   257  ;;;         SENSETAB       ch$c_ph_manage_ena       ; MANAGEMENT_ENABLED
015D   258
015D   259  char_type = logical
015D   260
015D   261              SENSETAB       ch$c_lg_mode_writing     ; MODE_WRITING_ALLOWED
015D   262              SENSETAB       ch$c_lg_term_bits        ; TERMINAL_TYPE
015D   263              SENSETAB       ch$c_lg_term_type        ; TERMINAL_SUBTYPE
015D   264              SENSETAB       ch$c_lg_output_flow      ; OUTPUT_FLOW_CONTROL
015D   265              SENSETAB       ch$c_lg_page_stop        ; OUTPUT_PAGE_STOP
015D   266              SENSETAB       ch$c_lg_flow_char_pass   ; FLOW_CHAR_PASS_THRU
015D   267              SENSETAB       ch$c_lg_input_flow       ; INPUT_FLOW_CONTROL
015D   268              SENSETAB       ch$c_lg_loss_notif       ; LOSS_NOTIFICATION
015D   269              SENSETAB       ch$c_lg_line_width       ; LINE_WIDTH
015D   270              SENSETAB       ch$c_lg_page_length      ; PAGE_LENGTH
015D   271              SENSETAB       ch$c_lg_stop_length      ; STOP_LENGTH
015D   272              SENSETAB       ch$c_lg_cr_fill          ; CR_FILL
015D   273              SENSETAB       ch$c_lg_lf_fill          ; LF_FILL
015D   274              SENSETAB       ch$c_lg_wrap             ; WRAP
015D   275              SENSETAB       ch$c_lg_hor_tab          ; HORIZONTAL_TAB
015D   276              SENSETAB       ch$c_lg_vert_tab         ; VERTICAL_TAB
015D   277              SENSETAB       ch$c_lg_form_feed        ; FORM_FEED
015D   278
015D   279  char_type = cterm
015D   280
015D   281              SENSETAB       ch$c_ct_ignore_input     ; IGNORE_INPUT
015D   282  ;;;         SENSETAB       ch$c_ct_char_att         ; CHAR_ATTRIBUTES
015D   283              SENSETAB       ch$c_ct_ctrlo_pass       ; CONTROL_O_PASS_THRU
015D   284              SENSETAB       ch$c_ct_raise_input      ; RAISE_INPUT
015D   285              SENSETAB       ch$c_ct_normal_echo      ; NORMAL_ECHO
015D   286              SENSETAB       ch$c_ct_input_esc        ; INPUT_ESCAPE_ENABLE
015D   287  ;;;         SENSETAB       ch$c_ct_output_esc       ; OUTPUT_ESCAPE_ENABLE
015D   288              SENSETAB       ch$c_ct_input_count      ; INPUT_COUNT_STATE
015D   289              SENSETAB       ch$c_ct_auto_prompt      ; AUTO_PROMPT
015D   290              SENSETAB       ch$c_ct_error_processing ; ERROR_PROCESSING
015D   291
015D   292  CTP$K_SENSEBUF == .-CTP$AB_SENSEBUF                 ; Size
015D   293
```

```
                        015D    294 .ENDC
                        015D    295 .nlist meb
                        015D    296
```

```
CT
Sy

TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
TT
UC
UN
VE
WR



PS
--
.
$A
RT



Ph
--
In
Co
Pa
Sy
Pa
Sy
Ps
Cr
As

Th
87
```

```
                        015D  .SBTTL  CT_POST_SENSE - Map TSA into VMS
                        015D
                        015D  ;
                        015D  ; Must map entire set of characteristics returned into VMS sense mode data
                        015D  ;
                        015D
                        015D  ; Input:
                        015D  ;       R2 - Address of CTP
                        015D  ;       R9 - Address of 12 byte buffer
                        015D  ;       R10 - Address of 8 byte buffer
                        015D  ;
                        015D  ; Output:
                        015D  ;       0(R9) - first longword of sense mode data
                        015D  ;       4(R9) - first longword of sense mode characteristics
                        015D  ;       8(R9) - second longword of sense mode characteristics
                        015D  ;
                        015D  ;       0(R10) - first longword of status
                        015D  ;       4(R10) - second longword of status
                        015D  ;
                        015D  ; Characteristics returned: (R9)
                        015D  ;
                        015D  ;       +-------+-------+-------+-------+
                        015D  ;       |  page width   | type  | class |
                        015D  ;       +-------+-------+-------+-------+
                        015D  ;       |length |     characteristics   |
                        015D  ;       +-------+-------+-------+-------+
                        015D  ;       |         characteristics       |
                        015D  ;       +-------+-------+-------+-------+
                        015D  ;
                        015D  ; IOSB: (R10)
                        015D  ;       +-------+-------+-------+-------+
                        015D  ;       |R speed|T speed|    status     |
                        015D  ;       +-------+-------+-------+-------+
                        015D  ;       |   0   |parity |LF fill|CR fill|
                        015D  ;       +-------+-------+-------+-------+
                        015D  ;
                        015D  ;
                        015D  ;
                        015D
                        015D  .IF DF RTPAD
                        015D  CT_CHAR_MSG::                           ; RTPAD ENTRY
                        015D  .IFF
                        015D  CT_POST_SENSE::                         ; CTDRIVER ENTRY
                        015D  .ENDC
                        015D
   51    28 A2    3C    015D          MOVZWL  CTP$W_MSGSIZE(R2),R1    ; Fetch size of buffer
   52    2C A2    9E    0161          MOVAB   CTP$W_CH_PARAM(R2),R2   ; Set address of first characteristic
7E    52    51    C1    0165          ADDL3   R1,R2,-(SP)             ; Save end address on stack
      6E    03    C2    0169          SUBL2   #3,(SP)         ; *** fudge, check this **
                        016C
   69    42 8F    90    016C          MOVB    #DC$_TERM,TT_BUF(R9)    ; Set terminal class
      6A    01    B0    0170          MOVW    #SS$_NORMAL,TT_IOSB(R10); Set status
                        0173
                        0173  10$:
      6E    52    D1    0173          CMPL    R2,(SP)                 ; Hit end of list yet?
            15    18    0176          BGEQ    25$                     ; Branch if so
      56    82    3C    0178          MOVZWL  (R2)+,R6                ; Get parameter
   57  FE81 CF    9E    017B          MOVAB   SENSE_TABLE,R7          ; Get table address
```

```
                                 0180   355 20$:
             87   56   B1   0180   356            CMPW    R6,(R7)+               ; Compare to table entry
                  0D   13   0183   357            BEQL    30$                    ; Branch if match found
                  87   B5   0185   358            TSTW    (R7)+                  ; Advance pointer past routine
                  F7   18   0187   359            BGEQ    20$                    ; Loop if greater than or equal to zero
                            0189   360
                            0189   361            ; PANIC                        ; 'invalid sense mode returned'
                            0189   362            MINOR_ERROR                    ; Increment error count, then exit
                            018D   363 25$:
                  8E   D5   018D   364            TSTL    (SP)+                  ; throw away end address
                02C3   31   018F   365            BRW     POST_SENSE_EXIT        ; exit code
                            0192   366 ;
                            0192   367 ; Dispatch to routine
                            0192   368 ;
                            0192   369 30$:
    56    000001A3'EF  9E   0192   370            MOVAB   SENSE_ROUTINES,R6      ; Base
             57   67   3C   0199   371            MOVZWL  (R7),R7                ; get routine address
             57   56   C0   019C   372            ADDL2   R6,R7                  ; Add offset to base for routine address
                  67   16   019F   373            JSB     (R7)                   ; jsb to routine
                  D0   11   01A1   374            BRB     10$                    ; Do next parameter
                            01A3   375
                            01A3   376 ;
                            01A3   377 ; All these routines have R0,R1,R4,R6-R8 as scratch
                            01A3   378 ;
                            01A3   379
```

```
                           01A3   381 SENSE_ROUTINES:
                           01A3   382
                           01A3   383 ; physical
                           01A3   384
                           01A3   385 INPUT_SPEED:                                  ; physical 1
          50     82    B0  01A3   386           MOVW    (R2)+,R0                   ; Get speed
                 0F    10  01A6   387           BSBB    GET_SPEED                  ; map into TT$C_BAUD_xxxx
       03 AA     50    90  01A8   388           MOVB    R0,TT_IOSB+3(R10)          ; set receive speed
                       05  01AC   389           RSB                                ; Return
                           01AD   390
                           01AD   391 OUTPUT_SPEED:                                 ; physical 2
          50     82    B0  01AD   392           MOVW    (R2)+,R0                   ; Get speed
                 05    10  01B0   393           BSBB    GET_SPEED                  ; map into TT$C_BAUD_xxxx
       02 AA     50    90  01B2   394           MOVB    R0,TT_IOSB+2(R10)          ; set transmit speed
                       05  01B6   395           RSB                                ; Return
                           01B7   396 ;
                           01B7   397 ; Local routine to map speed into TT$C_BAUD rates
                           01B7   398 ;
                           01B7   399 ; Input:
                           01B7   400 ;         R0 - baud rate in BPS
                           01B7   401 ;
                           01B7   402 ; Output:
                           01B7   403 ;         R0 - Baud rate in TT$C_BAUD_xxx terms
                           01B7   404 ;
                           01B7   405 GET_SPEED:
      51    FEDB CF    9E  01B7   406           MOVAB   SENSE_BAUD,R1              ; Get table
                           01BC   407 110$:
          81     50    B1  01BC   408           CMPW    R0,(R1)+                   ; compare
                 08    15  01BF   409           BLEQ    120$
                 81    95  01C1   410           TSTB    (R1)+                      ; Advance
                 F7    12  01C3   411           BNEQ    110$                       ; Branch if not zero
          50     10    90  01C5   412           MOVB    #TT$C_BAUD_19200,R0        ; Assume 19200 if > 19200
                 05        01C8   413           RSB                                ; Return
          50     61    90  01C9   414 120$:     MOVB    (R1),R0                    ; Fetch baud rate symbol value
                 05        01CC   415           RSB                                ; Return
                           01CD   416
                           01CD   417 CHARACTER_SIZE:                               ; physical 3
                           01CD   418 .IF DF RTPAD
   00008000 8F         CA  01CD   419           BICL    #TT$M_EIGHTBIT,-
          04 A9            01D3   420                   TT_CHAR1(R9)               ; Set characteristic
                           01D5   421 .ENDC
          82     07    B1  01D5   422           CMPW    #7,(R2)+                   ; Character size
                 08    18  01D8   423           BGEQ    10$                        ; exit if less than or equal to 7
   00008000 8F         C8  01DA   424           BISL    #TT$M_EIGHTBIT,-
          04 A9            01E0   425                   TT_CHAR1(R9)               ; Set characteristic
                       05  01E2   426 10$:      RSB                                ; Return
                           01E3   427
                           01E3   428 PARITY_ENABLE:                                ; physical 4
                           01E3   429 .IF DF RTPAD
       40 8F           8A  01E3   430           BICB    #TT$M_PARITY,-
       06 AA               01E6   431                   TT_IOSB+6(R10)             ; Set parity enabled
                           01E8   432 .ENDC
          50     82    90  01E8   433           MOVB    (R2)+,R0                   ; Get boolean
       05 50           E9  01EB   434           BLBC    R0,10$                     ; Branch if not enabled
       40 8F           88  01EE   435           BISB    #TT$M_PARITY,-
       06 AA               01F1   436                   TT_IOSB+6(R10)             ; Set parity enabled
                       05  01F3   437 10$:      RSB                                ; Return
```

```
                                 01F4   438
                                 01F4   439 PARITY_TYPE:                                        ; physical 5
                                 01F4   440 .IF DF RTPAD
                  80 8F     8A   01F4   441             BICB    #TT$M_ODD,-
                  06 AA          01F7   442                     TT_IOSB+6(R10)                  ; Assume even parity
                                 01F9   443 .ENDC
               50    82     3C   01F9   444             MOVZWL  (R2)+,R0                        ; Get parity number
               50    02     B1   01FC   445             CMPW    #ch$c_parity_odd,R0             ; Odd?
                     05     12   01FF   446             BNEQ    10$                             ; If not, exit
                  80 8F     88   0201   447             BISB    #TT$M_ODD,-
                  06 AA          0204   448                     TT_IOSB+6(R10)                  ; Set odd parity
                            05   0206   449 10$:        RSB                                     ; Return
                                 0207   450
                                 0207   451 MODEM_PRESENT:                                      ; physical 6
                                 0207   452 .IF DF RTPAD
     04 A9     00202000 8F      CA   0207   453             BICL    #<TT$M_MODEM!-
                                 020F   454                     TT$M_REMOTE>,TT_CHAR1(R9) ; Set modem and remote bits
                                 020F   455 .ENDC
               50    82     90   020F   456             MOVB    (R2)+,R0                        ; Get Boolean
                  08 50     E9   0212   457             BLBC    R0,10$                          ; Branch if not enabled
     04 A9     00202000 8F      C8   0215   458             BISL    #<TT$M_MODEM!-
                                 021D   459                     TT$M_REMOTE>,TT_CHAR1(R9) ; Set modem and remote bits
                            05   021D   460 10$:        RSB                                     ; Return
                                 021E   461
                                 021E   462 AUTOBAUD_DETECT:                                    ; physical 7
                                 021E   463 .IF DF RTPAD
                     02     CA   021E   464             BICL    #TT2$M_AUTOBAUD,-
                  08 A9          0220   465                     TT_CHAR2(R9)                    ; Set autobaud
                                 0222   466 .ENDC
               50    82     90   0222   467             MOVB    (R2)+,R0                        ; Get Boolean
                  04 50     E9   0225   468             BLBC    R0,10$                          ; Branch if not enabled
                     02     C8   0228   469             BISL    #TT2$M_AUTOBAUD,-
                  08 A9          022A   470                     TT_CHAR2(R9)                    ; Set autobaud
                            05   022C   471 10$:        RSB                                     ; Return
                                 022D   472
                                 022D   473 MANAGEMENT_GUARANTEED:                              ; physical 8
               50    82     90   022D   474             MOVB    (R2)+,R0                        ; Get Boolean
                  00 50     E9   0230   475             BLBC    R0,10$                          ; Branch if not enabled
                            05   0233   476 10$:        RSB                                     ; Return
                                 0234   477
                                 0234   478 SWITCH_CHAR_1:                                      ; physical 9
                  0217      30   0234   479             BSBW    IGNORE_STRING                   ; Ignore this string data
                            05   0237   480             RSB                                     ; Return
                                 0238   481
                                 0238   482 SWITCH_CHAR_2:                                      ; physical 10
                  0213      30   0238   483             BSBW    IGNORE_STRING                   ; Ignore this string data
                            05   023B   484             RSB                                     ; Return
                                 023C   485
```

```
                          023C  ; logical
                          023C
                          023C  MODE_WRITING_ALLOWED:                        ; logical 1
         50   82    90    023C           MOVB    (R2)+,R0                    ; Get Boolean
           00 50    E9    023F           BLBC    R0,10$                      ; Branch if not enabled
                    05    0242  10$:     RSB                                 ; Return
                          0243
                          0243  TERMINAL_TYPE:                               ; logical 2
                          0243
                          0243  ; bit mask is:
                          0243  ;        0 - Unknown/known
                          0243  ;        1 - scope/hardcopy
                          0243
         50   82    3C    0243           MOVZWL  (R2)+,R0                    ; Get characteristic
   00001000   8F    C8    0246           BISL    #TT$M_SCOPE,-
           04 A9          024C                   TT_CHAR1(R9)                ; Set scope
      08 50   01    E0    024E           BBS     #CTP$V_CH_SCOPE,R0,10$      ; Branch if scope
   00001000   8F    CA    0252           BICL    #TT$M_SCOPE,-
           04 A9          0258                   TT_CHAR1(R9)                ; Clear scope
                          025A
                          025A  ; *** unknown?
                          025A
      04 50   00    E0    025A  10$:     BBS     #CTP$V_CH_KNOWN,R0,20$      ; Branch if known
      01 A9   00    90    025E           MOVB    #TT$_UNKNOWN,TT_BUF+1(R9)   ; Set terminal type
              05          0262  20$:     RSB                                 ; Return
                          0263
                          0263  TERMINAL_SUBTYPE:                            ; logical 3
         56   82    9A    0263           MOVZBL  (R2)+,R6                    ; Get length of string
              01    12    0266           BNEQ    10$                         ; Continue if not 0
                    05    0268           RSB                                 ; Return
                          0269  10$:
         57   52    D0    0269           MOVL    R2,R7                       ; Get address of string
         52   56    C0    026C           ADDL2   R6,R2                       ; Update pointer
       040C   8F    BB    026F           PUSHR   #^M<R2,R3,R10>              ; Save registers
   58  FE53   CF    9E    0273           MOVAB   CT$AB_TERM_TABLE,R8         ; Address of table
              54    D4    0278           CLRL    R4                          ; Zero length
                          027A  20$:
         58   54    C0    027A           ADDL2   R4,R8                       ; Get next entry
      01 A9   88    90    027D           MOVB    (R8)+,TT_BUF+1(R9)          ; Set terminal type
              0E    13    0281           BEQL    30$                         ; branch if end of list
         54   88    9A    0283           MOVZBL  (R8)+,R4                    ; Get length
         54   56    91    0286           CMPB    R6,R4                       ; compare lengths
              EF    12    0289           BNEQ    20$                         ; not equal, loop
   68    67   56    29    028B           CMPC3   R6,(R7),(R8)                ; Compare
              E9    12    028F           BNEQ    20$                         ; Loop if not equal
                          0291
                          0291  ; Terminal type match
                          0291
                          0291           ASSUME  TT$_UNKNOWN EQ 0            ; assume for end of table code
                          0291  30$:
       040C 8F      BA    0291           POPR    #^M<R2,R3,R10>              ; Restore registers
                          0295
                    05    0295           RSB                                 ; Return
                          0296
                          0296
                          0296  OUTPUT_FLOW_CONTROL:                         ; logical 4
                          0296  .IF DF RTPAD
              20   CA  0296   544              BICL    #TT$M_TTSYNC,-
           04 A9       0298   545                      TT_CHAR1(R9)               ; Set tt sync
                       029A   546      .ENDC
           50 82   90  029A   547              MOVB    (R2)+,R0                   ; Get Boolean
           04 50   E9  029D   548              BLBC    R0,10$                     ; Branch if not enabled
              20   C8  02A0   549              BISL    #TT$M_TTSYNC,-
           04 A9       02A2   550                      TT_CHAR1(R9)               ; Set tt sync
              05       02A4   551      10$:    RSB                                ; Return
                       02A5   552
                       02A5   553      OUTPUT_PAGE_STOP:                          ; logical 5
                       02A5   554      .IF DF RTPAD
     00004000 8F   CA  02A5   555              BICL    #TT$M_HOLDSCREEN,-
           04 A9       02AB   556                      TT_CHAR1(R9)               ; Set hold screen
                       02AD   557      .ENDC
           50 82   90  02AD   558              MOVB    (R2)+,R0                   ; Get Boolean
           08 50   E9  02B0   559              BLBC    R0,10$                     ; Branch if not enabled
     00004000 8F   C8  02B3   560              BISL    #TT$M_HOLDSCREEN,-
           04 A9       02B9   561                      TT_CHAR1(R9)               ; Set hold screen
              05       02BB   562      10$:    RSB                                ; Return
                       02BC   563
                       02BC   564      FLOW_CHAR_PASS_THRU:                       ; logical 6
           50 82   90  02BC   565              MOVB    (R2)+,R0                   ; Get Boolean
           00 50   E9  02BF   566              BLBC    R0,10$                     ; Branch if not enabled
              05       02C2   567      10$:    RSB                                ; Return
                       02C3   568
                       02C3   569      INPUT_FLOW_CONTROL:                        ; logical 7
                       02C3   570      .IF DF RTPAD
              10   CA  02C3   571              BICL    #TT$M_HOSTSYNC,-
           04 A9       02C5   572                      TT_CHAR1(R9)               ; Set hostsync
                       02C7   573      .ENDC
           50 82   90  02C7   574              MOVB    (R2)+,R0                   ; Get Boolean
           04 50   E9  02CA   575              BLBC    R0,10$                     ; Branch if not enabled
              10   C8  02CD   576              BISL    #TT$M_HOSTSYNC,-
           04 A9       02CF   577                      TT_CHAR1(R9)               ; Set hostsync
              05       02D1   578      10$:    RSB                                ; Return
                       02D2   579
                       02D2   580      LOSS_NOTIFICATION:                         ; logical 8
           50 82   90  02D2   581              MOVB    (R2)+,R0                   ; Get Boolean
           00 50   E9  02D5   582              BLBC    R0,10$                     ; Branch if not enabled
              05       02D8   583      10$:    RSB                                ; Bell on loss data?
                       02D9   584
                       02D9   585      LINE_WIDTH:                                ; logical 9
     02 A9    82   B0  02D9   586              MOVW    (R2)+,TT_BUF+2(R9)         ; Line width
              05       02DD   587              RSB                                ; Return
                       02DE   588
                       02DE   589      PAGE_LENGTH:                               ; logical 10
     07 A9    82   33  02DE   590              CVTWB   (R2)+,TT_CHAR1+3(R9)       ; Page length
              05       02E2   591              RSB                                ; Return
                       02E3   592
                       02E3   593      STOP_LENGTH:                               ; logical 11
              82   B5  02E3   594              TSTW    (R2)+                      ; Ignore for now
              05       02E5   595              RSB                                ; Return
                       02E6   596
                       02E6   597      CR_FILL:                                   ; logical 12
                       02E6   598      .IF DF RTPAD
     00000400 8F   CA  02E6   599              BICL    #TT$M_CRFILL,-
           04 A9       02EC   600                      TT_CHAR1(R9)
```

```
                        02EE   .ENDC
   04 AA     82   33    02EE               CVTWB   (R2)+,TT_IOSB+4(R10)
             08   13    02F2               BEQL    10$                    ; Branch if zero
00000400 8F       C8    02F4               BISL    #TT$M_CRFILL,-
      04 A9             02FA                       TT_CHAR1(R9)
                  05    02FC   10$:        RSB                            ; Return
                        02FD
                        02FD   LF_FILL:                                   ; logical 13
                        02FD   .IF DF RTPAD
00000800 8F       CA    02FD               BICL    #TT$M_LFFILL,-
      04 A9             0303                       TT_CHAR1(R9)
                        0305   .ENDC
   05 AA     82   33    0305               CVTWB   (R2)+,TT_IOSB+5(R10)
             08   13    0309               BEQL    10$                    ; Branch if zero
00000800 8F       C8    030B               BISL    #TT$M_LFFILL,-
      04 A9             0311                       TT_CHAR1(R9)
                  05    0313   10$:        RSB                            ; Return
                        0314
                        0314   WRAP:                                      ; logical 14
                        0314   .IF DF RTPAD
00000200 8F       CA    0314               BICL    #TT$M_WRAP,-
      04 A9             031A                       TT_CHAR1(R9)           ; set wrap
                        031C   .ENDC
        82   03   B1    031C               CMPW    #3,(R2)+               ; Get wrap (value 1 to 4)
             08   18    031F               BGEQ    10$                    ; if 3 or less, no wrap
00000200 8F       C8    0321               BISL    #TT$M_WRAP,-
      04 A9             0327                       TT_CHAR1(R9)           ; set wrap
                  05    0329   10$:        RSB                            ; Return
                        032A
                        032A   HORIZONTAL_TAB:                            ; logical 15
                        032A   .IF DF RTPAD
00000100 8F       CA    032A               BICL    #TT$M_MECHTAB,-
      04 A9             0330                       TT_CHAR1(R9)           ; Set mechtab
                        0332   .ENDC
        82   01   B1    0332               CMPW    #1,(R2)+               ; mechtab?
             08   12    0335               BNEQ    10$                    ; Branch if no
00000100 8F       C8    0337               BISL    #TT$M_MECHTAB,-
      04 A9             033D                       TT_CHAR1(R9)           ; Set mechtab
                  05    033F   10$:        RSB                            ; Return
                        0340
                        0340   VERTICAL_TAB:                              ; logical 16
             82   B5    0340               TSTW    (R2)+                  ; ignore
                  05    0342               RSB                            ; Return
                        0343
                        0343   FORM_FEED:                                 ; logical 17
                        0343   .IF DF RTPAD
00080000 8F       CA    0343               BICL    #TT$M_MECHFORM,-
      04 A9             0349                       TT_CHAR1(R9)           ; Set mechform
                        034B   .ENDC
        82   01   B1    034B               CMPW    #1,(R2)+               ; mech form?
             08   12    034E               BNEQ    10$                    ; Branch if no
00080000 8F       C8    0350               BISL    #TT$M_MECHFORM,-
      04 A9             0356                       TT_CHAR1(R9)           ; Set mechform
                  05    0358   10$:        RSB                            ; Return
```

```
                      0359  ; cterm
                      0359
                      0359  IGNORE_INPUT:                                 ; cterm 1
         50   82  90  0359              MOVB    (R2)+,R0                  ; Get Boolean
           00 50  E9  035C              BLBC    R0,10$                    ; Branch if not enabled
                  05  035F  10$:        RSB                               ; Return
                      0360
                      0360  CHAR_ATTRIBUTES:                              ; cterm 2
                      0360
                      0360  .IF DF RTPAD
                      0360
50   00000000'EF  9E  0360              MOVAB   OUTBAND_NEW,R0            ; Get temporary mask
         51   82  9A  0367              MOVZBL  (R2)+,R1                  ; get character
         56   82  9A  036A              MOVZBL  (R2)+,R6                  ; Get mask
         57   82  9A  036D              MOVZBL  (R2)+,R7                  ; Get attribute
              54  D4  0370              CLRL    R4                        ; Clear
      00 54   51  E2  0372              BBSS    R1,R4,5$                  ; turn character into mask
                      0376  5$:
              00  EF  0376              EXTZV   #CTP$V_CH_00,-
    58   56   02      0378                      #CTP$S_CH_00,R6,R8        ; Fetch isolate mask
         58   03  91  037B              CMPB    #CTP$M_CH_00,R8           ; Must be full mask
              3F  12  037E              BNEQ    100$                      ; exit *** - other things to handle here
                      0380
              00  EF  0380              EXTZV   #CTP$V_CH_00,-
    58   57   02      0382                      #CTP$S_CH_00,R7,R8        ; Fetch attributes
         58   00  91  0385              CMPB    #CTP$C_CH_CANCEL,R8       ; Cancel?
              0E  12  0388              BNEQ    10$                       ;  br if no
      04 A0   54  CA  038A              BICL    R4,OOB_EXCLUDE(R0)        ; clear them all...
      0C A0   54  CA  038E              BICL    R4,OOB_INCLUDE(R0)
      14 A0   54  CA  0392              BICL    R4,OOB_ABORT(R0)
              27  11  0396              BRB     100$
                      0398  10$:
         58   01  91  0398              CMPB    #CTP$C_CH_ICLEAR,R8       ; immediate clear?
              05  13  039B              BEQL    20$                       ; br if yes
         58   02  91  039D              CMPB    #CTP$C_CH_DCLEAR,R8       ; deferred clear?
              06  12  03A0              BNEQ    30$                       ; br if no
                      03A2  20$:
                      03A2              ;
                      03A2              ; set up abort out of band
                      03A2              ;
      14 A0   54  C8  03A2              BISL    R4,OOB_ABORT(R0)          ; Set abort flag
              17  11  03A6              BRB     100$
                      03A8  30$:
         58   03  91  03A8              CMPB    #CTP$C_CH_HELLO,R8        ; hello?
              12  12  03AB              BNEQ    100$                      ; no, (sanity check really)
      0A 56   02  E1  03AD              BBC     #CTP$V_CH_I,R6,40$        ; Branch if include not specified
      06 57   02  E1  03B1              BBC     #CTP$V_CH_I,R7,40$        ; Branch if include not required
      0C A0   54  C8  03B5              BISL    R4,OOB_INCLUDE(R0)        ; Set include bit
              04  11  03B9              BRB     100$
                      03BB  40$:
      04 A0   54  C8  03BB              BISL    R4,OOB_EXCLUDE(R0)        ; set exclude bit
                      03BF  ;
                      03BF  ; Handle out-of-band discard                  (D)
                      03BF  ;
                      03BF  100$:
      0C 56   03  E1  03BF              BBC     #CTP$V_CH_D,R6,200$       ; Skip if not in select mask
      18 A0   54  CA  03C3              BICL    R4,OOB_DISCARD(R0)        ; Assume no discard output
```

```
           04 57    03   E1  03C7             BBC     #CTP$V_CH_D,R7,200$              ; Skip if not in select mask
           18 A0    54   C8  03CB             BISL    R4,OOB_DISCARD(R0)               ; Set discard output
                             03CF      200$:
                             03CF      ;
                             03CF      ; Handle control character echoing       (EE)
                             03CF      ;
                    04   EF  03CF             EXTZV   #CTP$V_CH_EE,-
        58    56    02       03D1                     #CTP$S_CH_EE,R6,R8               ; Fetch isolate mask
              58    03   91  03D4             CMPB    #3,R8                            ; specified?
                    17   12  03D7             BNEQ    300$                             ; nope
           1C A0    54   CA  03D9             BICL    R4,OOB_ECHO(R0)                  ; assume no echo (like ^T)
                    04   EF  03DD             EXTZV   #CTP$V_CH_EE,-
        58    57    02       03DF                     #CTP$S_CH_EE,R7,R8               ; Fetch real data
              58    00   91  03E2             CMPB    #CTP$C_CH_ECHONONE,R8            ; echo none?
                    09   13  03E5             BEQL    300$                             ; yes, continue
              58    02   91  03E7             CMPB    #CTP$C_CH_ECHOSTANDARD,R8              ; echo STANDARD?
                    04   12  03EA             BNEQ    300$                             ; branch if other...
           1C A0    54   C8  03EC             BISL    R4,OOB_ECHO(R0)                  ; set standard echo  (like ^C)
                             03F0      300$:
                             03F0      ;
                             03F0      ; *** code not done for:
                             03F0      ;                  enable/disable special character (F)
                             03F0
                    05       03F0             RSB                                      ; Exit
                             03F1
                             03F1      .IFF       ; CTDRIVER
                             03F1
                             03F1             ADDL    #3,R2                            ; Skip (should never really get here)
                             03F1             RSB
                             03F1
                             03F1      .ENDC
                             03F1
                             03F1      CONTROL_O_PASS_THRU:                            ; cterm 3
           50  82       90   03F1             MOVB    (R2)+,R0                         ; Get Boolean
           00  50       E9   03F4             BLBC    R0,10$                           ; Branch if not enabled
                    05       03F7      10$:   RSB                                      ; Return
                             03F8
                             03F8      RAISE_INPUT:                                    ; cterm 4
                             03F8      .IF DF RTPAD
      00000080 8F       CA   03F8             BICL    #TT$M_LOWER,-
           04 A9             03FE                     TT_CHAR1(R9)                     ; Set convert lower
                             0400      .ENDC
           50  82       90   0400             MOVB    (R2)+,R0                         ; Get Boolean
           08  50       E8   0403             BLBS    R0,10$                           ; Branch if not enabled
      00000080 8F       C8   0406             BISL    #TT$M_LOWER,-
           04 A9             040C                     TT_CHAR1(R9)                     ; Set convert lower
                    05       040E      10$:   RSB                                      ; Return
                             040F
                             040F      NORMAL_ECHO:                                    ; cterm 5
                             040F      .IF DF RTPAD
                 02     CA   040F             BICL    #TT$M_NOECHO,-
           04 A9             0411                     TT_CHAR1(R9)                     ; Set noecho
                             0413      .ENDC
           50  82       90   0413             MOVB    (R2)+,R0                         ; Get Boolean
           04  50       E8   0416             BLBS    R0,10$                           ; Branch if enabled (note opposite sense)
                 02     C8   0419             BISL    #TT$M_NOECHO,-
           04 A9             041B                     TT_CHAR1(R9)                     ; Set noecho
```

```
                   05  041D   770 10$:     RSB                                    ; Return
                       041E   771
                       041E   772 INPUT_ESCAPE_ENABLE:                            ; cterm 6
                       041E   773 OUTPUT_ESCAPE_ENABLE:                           ; cterm 7
                       041E   774 .IF DF RTPAD
             08    CA  041E   775          BICL    #TT$M_ESCAPE,-
          04 A9        0420   776                  TT_CHAR1(R9)
                       0422   777 .ENDC
       50    82    90  0422   778          MOVB    (R2)+,R0               ; Get Boolean
          04 50    E9  0425   779          BLBC    R0,10$                 ; Branch if not enabled
             08    C8  0428   780          BISL    #TT$M_ESCAPE,-
          04 A9        042A   781                  TT_CHAR1(R9)
                   05  042C   782 10$:     RSB                                    ; Return
                       042D   783
                       042D   784 INPUT_COUNT_STATE:                              ; cterm 8
             82    B5  042D   785          TSTW    (R2)+                  ; Ignore
                   05  042F   786          RSB
                       0430   787
                       0430   788 AUTO_PROMPT:                                    ; cterm 9
                       0430   789 .IF DF RTPAD
  00000040 8F      CA  0430   790          BICL    #TT$M_SCRIPT,-
          04 A9        0436   791                  TT_CHAR1(R9)
                       0438   792 .ENDC
       50    82    90  0438   793          MOVB    (R2)+,R0               ; Get Boolean
          08 50    E9  043B   794          BLBC    R0,10$                 ; Branch if not enabled
  00000040 8F      C8  043E   795          BISL    #TT$M_SCRIPT,-
          04 A9        0444   796                  TT_CHAR1(R9)
                   05  0446   797 10$:     RSB                                    ; Return
                       0447   798
                       0447   799 ERROR_PROCESSING:                               ; cterm 10
       50    82    90  0447   800          MOVB    (R2)+,R0               ; Get Boolean
          00 50    E9  044A   801          BLBC    R0,10$                 ; Branch if not enabled
                   05  044D   802 10$:     RSB                                    ; Return
                       044E   803
                       044E   804
                       044E   805 IGNORE_STRING:
       50    82    9A  044E   806          MOVZBL  (R2)+,R0               ; Get length of string
       52    50    C0  0451   807          ADDL2   R0,R2                  ; Add to address
                   05  0454   808          RSB
                       0455   809
```

```
        0455   811 POST_SENSE_EXIT:
        0455   812
     05 0455   813           RSB
        0456   814
        0456   815 .end
```

```
                                                 B 14
CTSETRT                    - RTPAD/CTERM SET CHARACTERISTICS     16-SEP-1984 02:11:58  VAX/VMS Macro V04-00        Page  19
Symbol table                                                      5-SEP-1984 03:14:35  [RTPAD.SRC]CTDSENSE.MAR;1        (7)

AUTOBAUD_DETECT           0000021E R    02    OUTBAND_NEW               *********  X  02
AUTO_PROMPT               00000430 R    02    OUTPUT_ESCAPE_ENABLE      0000041E R    02
CH$C_CTERM              = 00000002            OUTPUT_FLOW_CONTROL       00000296 R    02
CH$C_LOGICAL            = 00000001            OUTPUT_PAGE_STOP          000002A5 R    02
CH$C_PARITY_ODD         = 00000002            OUTPUT_SPEED              000001AD R    02
CH$C_PHYSICAL           = 00000000            PAGE_LENGTH               000002DE R    02
CHARACTER_SIZE            000001CD R    02    PARITY_ENABLE             000001E3 R    02
CHAR_ATTRIBUTES           00000360 R    02    PARITY_TYPE               000001F4 R    02
CHAR_COUNT              = 0000000B            PHYSICAL                = 00000000
CHAR_TYPE               = 00000200            POST_SENSE_EXIT           00000455 R    02
CONTROL_O_PASS_THRU       000003F1 R    02    RAISE_INPUT               000003F8 R    02
CR_FILL                   000002E6 R    02    RTPAD                   = 00000001
CT$AB_TERM_TABLE          000000CA RG   02    SENSE_BAUD                00000096 R    02
CTERM                   = 00000200            SENSE_ROUTINES            000001A3 R    02
CTP$C_CH_CANCEL         = 00000000            SENSE_TABLE               00000000 R    02
CTP$C_CH_DCLEAR         = 00000002            SS$_NORMAL              = 00000001
CTP$C_CH_ECHONONE       = 00000000            STOP_LENGTH               000002E3 R    02
CTP$C_CH_ECHOSTANDARD   = 00000002            SWITCH_CHAR_1             00000234 R    02
CTP$C_CH_HELLO          = 00000003            SWITCH_CHAR_2             00000238 R    02
CTP$C_CH_ICLEAR         = 00000001            TERMINAL_SUBTYPE          00000263 R    02
CTP$M_CH_OO             = 00000003            TERMINAL_TYPE             00000243 R    02
CTP$S_CH_EE             = 00000002            TT$C_BAUD_110           = 00000003
CTP$S_CH_OO             = 00000002            TT$C_BAUD_1200          = 00000008
CTP$V_CH_D              = 00000003            TT$C_BAUD_134           = 00000004
CTP$V_CH_EE             = 00000004            TT$C_BAUD_150           = 00000005
CTP$V_CH_I              = 00000002            TT$C_BAUD_1800          = 00000009
CTP$V_CH_KNOWN          = 00000000            TT$C_BAUD_19200         = 00000010
CTP$V_CH_OO             = 00000000            TT$C_BAUD_2000          = 0000000A
CTP$V_CH_SCOPE          = 00000001            TT$C_BAUD_2400          = 0000000B
CTP$W_CH_PARAM          = 0000002C            TT$C_BAUD_300           = 00000006
CTP$W_MSGSIZE           = 00000028            TT$C_BAUD_3600          = 0000000C
CT_CHAR_MSG               0000015D RG   02    TT$C_BAUD_4800          = 0000000D
DC$_TERM                = 00000042            TT$C_BAUD_50            = 00000001
ERROR_PROCESSING          00000447 R    02    TT$C_BAUD_600           = 00000007
FLOW_CHAR_PASS_THRU       000002BC R    02    TT$C_BAUD_7200          = 0000000E
FORM_FEED                 00000343 R    02    TT$C_BAUD_75            = 00000002
GET_SPEED                 000001B7 R    02    TT$C_BAUD_9600          = 0000000F
HORIZONTAL_TAB            0000032A R    02    TT$M_CRFILL             = 00000400
IGNORE_INPUT              00000359 R    02    TT$M_EIGHTBIT           = 00008000
IGNORE_STRING             0000044E R    02    TT$M_ESCAPE             = 00000008
INPUT_COUNT_STATE         0000042D R    02    TT$M_HOLDSCREEN         = 00004000
INPUT_ESCAPE_ENABLE       0000041E R    02    TT$M_HOSTSYNC           = 00000010
INPUT_FLOW_CONTROL        000002C3 R    02    TT$M_LFFILL             = 00000800
INPUT_SPEED               000001A3 R    02    TT$M_LOWER              = 00000080
LF_FILL                   000002FD R    02    TT$M_MECHFORM           = 00080000
LINE_WIDTH                000002D9 R    02    TT$M_MECHTAB            = 00000100
LOGICAL                 = 00000100            TT$M_MODEM              = 00200000
LOSS_NOTIFICATION         000002D2 R    02    TT$M_NOECHO             = 00000002
MANAGEMENT_GUARANTEED     0000022D R    02    TT$M_ODD                = 00000080
MODEM_PRESENT             00000207 R    02    TT$M_PARITY             = 00000040
MODE_WRITING_ALLOWED      0000023C R    02    TT$M_REMOTE             = 00002000
NORMAL_ECHO               0000040F R    02    TT$M_SCOPE              = 00001000
OOB_ABORT               = 00000014            TT$M_SCRIPT             = 00000040
OOB_DISCARD             = 00000018            TT$M_TTSYNC             = 00000020
OOB_ECHO                = 0000001C            TT$M_WRAP               = 00000200
OOB_EXCLUDE             = 00000004            TT$_LA100               = 00000025
OOB_INCLUDE             = 0000000C            TT$_LA12                = 00000024
```

```
TT$_LA24                        = 00000025
TT$_LA34                        = 00000022
TT$_LA36                        = 00000020
TT$_LA38                        = 00000023
TT$_LQP02                       = 00000026
TT$_UNKNOWN                     = 00000000
TT$_VK100                       = 00000002
TT$_VT05                        = 00000001
TT$_VT100                       = 00000060
TT$_VT101                       = 00000061
TT$_VT102                       = 00000062
TT$_VT105                       = 00000063
TT$_VT125                       = 00000064
TT$_VT131                       = 00000065
TT$_VT132                       = 00000066
TT$_VT173                       = 00000003
TT$_VT200_SERIES                = 0000006E
TT$_VT52                        = 00000040
TT$_VT55                        = 00000041
TT2$M_AUTOBAUD                  = 00000002
TT_BUF                          = 00000000
TT_CHAR1                        = 00000004
TT_CHAR2                        = 00000008
TT_IOSB                         = 00000000
UCB$W_ERRCNT                    = 00000082
UNKNOWN_TT                        00000155 R     02
VERTICAL_TAB                      00000340 R     02
WRAP                              00000314 R     02

                                              +----------------+
                                              ! Psect synopsis !
                                              +----------------+

PSECT name                      Allocation            PSECT No.  Attributes
----------                      ----------            ---------  ----------
.  ABS  .                       00000000  (      0.)  00 (   0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                           00000000  (      0.)  01 (   1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
RTPAD                           00000456  (   1110.)  02 (   2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD   NOWRT NOVEC BYTE

                                           +------------------------+
                                           ! Performance indicators !
                                           +------------------------+

Phase                        Page faults    CPU Time       Elapsed Time
-----                        -----------    --------       ------------
Initialization                        34    00:00:00.03    00:00:01.27
Command processing                   123    00:00:00.59    00:00:02.97
Pass 1                               412    00:00:10.10    00:00:38.48
Symbol table sort                      4    00:00:01.74    00:00:05.09
Pass 2                               143    00:00:02.19    00:00:11.07
Symbol table output                   17    00:00:00.11    00:00:00.11
Psect synopsis output                  2    00:00:00.02    00:00:00.02
Cross-reference output                 0    00:00:00.00    00:00:00.00
Assembler run totals                 737    00:00:14.78    00:00:59.01

The working set limit was 1800 pages.
87574 bytes (172 pages) of virtual memory were used to buffer the intermediate code.
```

```
There were 90 pages of symbol table space allocated to hold 1619 non-local and 43 local symbols.
816 source lines were read in Pass 1, producing 16 object records in Pass 2.
19 pages of virtual memory were used to define 18 macros.

                                        +--------------------------+
                                        ! Macro library statistics !
                                        +--------------------------+

Macro library name                      Macros defined
------------------                      --------------
_$255$DUA28:[RTPAD.OBJ]RTPAD.MLB;1                   2
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                       1
_$255$DUA28:[SYSLIB]STARLET.MLB;2                    8
TOTALS (all libraries)                              11

1655 GETS were required to define 11 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:CTSETRT/OBJ=OBJ$:CTSETRT MSRC$:CTSETRT/UPDATE=(ENH$:CTSETRT)+MSRC$:CTDSENSE/UPDATE=(ENH$:CTDSENSE)+EXECML$/LIB+LIB$:R
```

0333 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY